



Large-Scale Machine Learning for Malware Characterization using Graphs



CYBER 20/20
Deep Machine Learning
Meets Cybersecurity





John Cavazos, Ph.D.

CEO and Founder of Cyber 20/20

- Current Associate Professor at the University of Delaware
- Former JP Morgan Faculty Fellow at Inst. For Financial Services Analytics
- 20 years experience applying machine learning to problems involving code
- Spent one year with High Performance Computing Group at JP Morgan



The Problem



- Over 100K malware variants created every hour
- Bad actors have embraced automation
- Good actors still construct security defenses manually
 - Slow and error prone

The Problem

- “Mis-match” between bad actor techniques and good actor tools has caused high profile breaches to occur:



WannaCry Ransomware Attack

Patch for Unsupported Windows (Apply Now)

Hackers accessed SWIFT to Steal \$81 Million & Erase Evidence

4. Confirmation messages from the SWIFT network are now monitored by the malware. Functionality continues in <https://www.swift.com/press/2016-02-05-01>





Today's Methods of Protection



Security Products

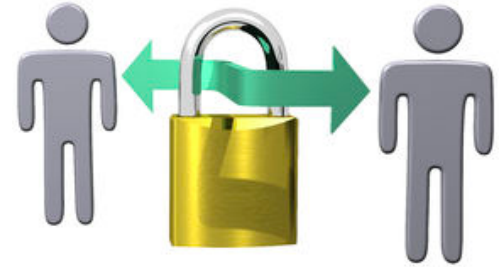
UPDATE



Software Patching



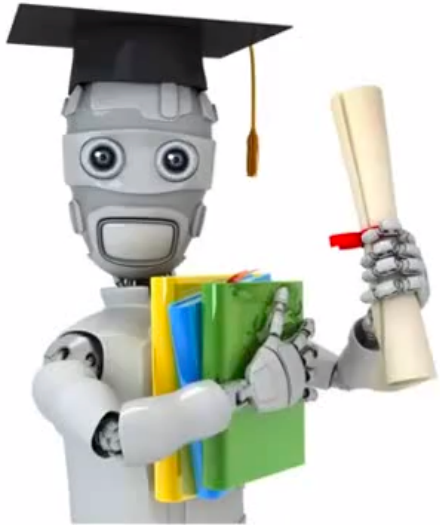
Employee Training



Secure Communication

But are these effective? Results say...not really!

Advanced Solutions Needed!



```

[00004038, 32, 185, 1] from 0x00000000 ip 0x00000000
0x00004038: 8b52ac21ffff mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x00004039: 8942d008      mov     dword [esp + 8], max ; copy & self exec as:
0x0000403a: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000403b: 8942d004      mov     dword [esp + 4], max ; fushbin@gigajimkidd
0x0000403c: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000403d: 83042400      mov     dword [esp], max
0x0000403e: e81fa1ffff      call    sym.writefile
0x0000403f: 83c00000      test    eax, eax ;(1) sym.writefile()
0x00004040: 741c0000      jz     0x00004049 ;(2)
0x00004041: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x00004042: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004043: 8942d008      mov     dword [esp + 8], max
0x00004044: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004045: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x00004046: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x00004047: 8b4dc000      mov     dword [ebp - 0x4c], max ; fushbin@gigajimkidd
0x00004048: e81fa1ffff      call    sym.writefile
0x00004049: 83c00000      test    eax, eax ;(1) sym.writefile()
0x0000404a: 741c0000      jz     0x00004059 ;(2)
0x0000404b: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x0000404c: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000404d: 8942d008      mov     dword [esp + 8], max
0x0000404e: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000404f: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x00004050: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x00004051: e81fa1ffff      call    sym.writefile
0x00004052: 83c00000      test    eax, eax ;(1) sym.writefile()
0x00004053: 741c0000      jz     0x00004062 ;(2)
0x00004054: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x00004055: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004056: 8942d008      mov     dword [esp + 8], max
0x00004057: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004058: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x00004059: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x0000405a: e81fa1ffff      call    sym.writefile
0x0000405b: 83c00000      test    eax, eax ;(1) sym.writefile()
0x0000405c: 741c0000      jz     0x0000406b ;(2)
0x0000405d: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x0000405e: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000405f: 8942d008      mov     dword [esp + 8], max
0x00004060: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004061: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x00004062: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x00004063: e81fa1ffff      call    sym.writefile
0x00004064: 83c00000      test    eax, eax ;(1) sym.writefile()
0x00004065: 741c0000      jz     0x00004074 ;(2)
0x00004066: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x00004067: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004068: 8942d008      mov     dword [esp + 8], max
0x00004069: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000406a: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x0000406b: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x0000406c: e81fa1ffff      call    sym.writefile
0x0000406d: 83c00000      test    eax, eax ;(1) sym.writefile()
0x0000406e: 741c0000      jz     0x0000407d ;(2)
0x0000406f: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x00004070: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004071: 8942d008      mov     dword [esp + 8], max
0x00004072: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004073: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x00004074: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x00004075: e81fa1ffff      call    sym.writefile
0x00004076: 83c00000      test    eax, eax ;(1) sym.writefile()
0x00004077: 741c0000      jz     0x00004086 ;(2)
0x00004078: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x00004079: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000407a: 8942d008      mov     dword [esp + 8], max
0x0000407b: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000407c: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x0000407d: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x0000407e: e81fa1ffff      call    sym.writefile
0x0000407f: 83c00000      test    eax, eax ;(1) sym.writefile()
0x00004080: 741c0000      jz     0x00004090 ;(2)
0x00004081: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x00004082: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004083: 8942d008      mov     dword [esp + 8], max
0x00004084: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004085: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x00004086: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x00004087: e81fa1ffff      call    sym.writefile
0x00004088: 83c00000      test    eax, eax ;(1) sym.writefile()
0x00004089: 741c0000      jz     0x00004099 ;(2)
0x0000408a: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x0000408b: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000408c: 8942d008      mov     dword [esp + 8], max
0x0000408d: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000408e: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x0000408f: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x00004090: e81fa1ffff      call    sym.writefile
0x00004091: 83c00000      test    eax, eax ;(1) sym.writefile()
0x00004092: 741c0000      jz     0x000040a1 ;(2)
0x00004093: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x00004094: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004095: 8942d008      mov     dword [esp + 8], max
0x00004096: 8b4dc000      mov     dword [ebp - 0x4c], max
0x00004097: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x00004098: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x00004099: e81fa1ffff      call    sym.writefile
0x0000409a: 83c00000      test    eax, eax ;(1) sym.writefile()
0x0000409b: 741c0000      jz     0x000040ab ;(2)
0x0000409c: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x0000409d: 8b4dc000      mov     dword [ebp - 0x4c], max
0x0000409e: 8942d008      mov     dword [esp + 8], max
0x0000409f: 8b4dc000      mov     dword [ebp - 0x4c], max
0x000040a0: 8352b0e0ffff      mov     eax, dword [ebp - 0xd354] ; this is where the malware
0x000040a1: 8942d004      mov     dword [esp + 4], max ; copy & self exec as:
0x000040a2: e81fa1ffff      call    sym.writefile
0x000040a3: 83c00000      test    eax, eax ;(1) sym.writefile()
0x000040a4: 741c0000      jz     0x000040b1 ;(2)
0x000040a5: 8b52ac21ffff      mov     eax, dword [ebp - 0xd354]
0x000040a6: 8b4dc000      mov     dword [ebp - 0x4c], max
0x000040a7: 8942d008      mov     dword [esp + 8], max
0x000040a8: 8b4dc000      mov     dword [ebp - 0x4c], max
0x000040a9: 83
```

Machine Learning

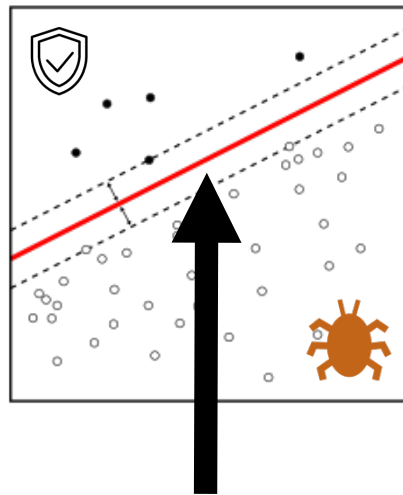


Malware Characterization

What is Machine Learning?

**Computer algorithms
that learn from and
make predictions on
data.**

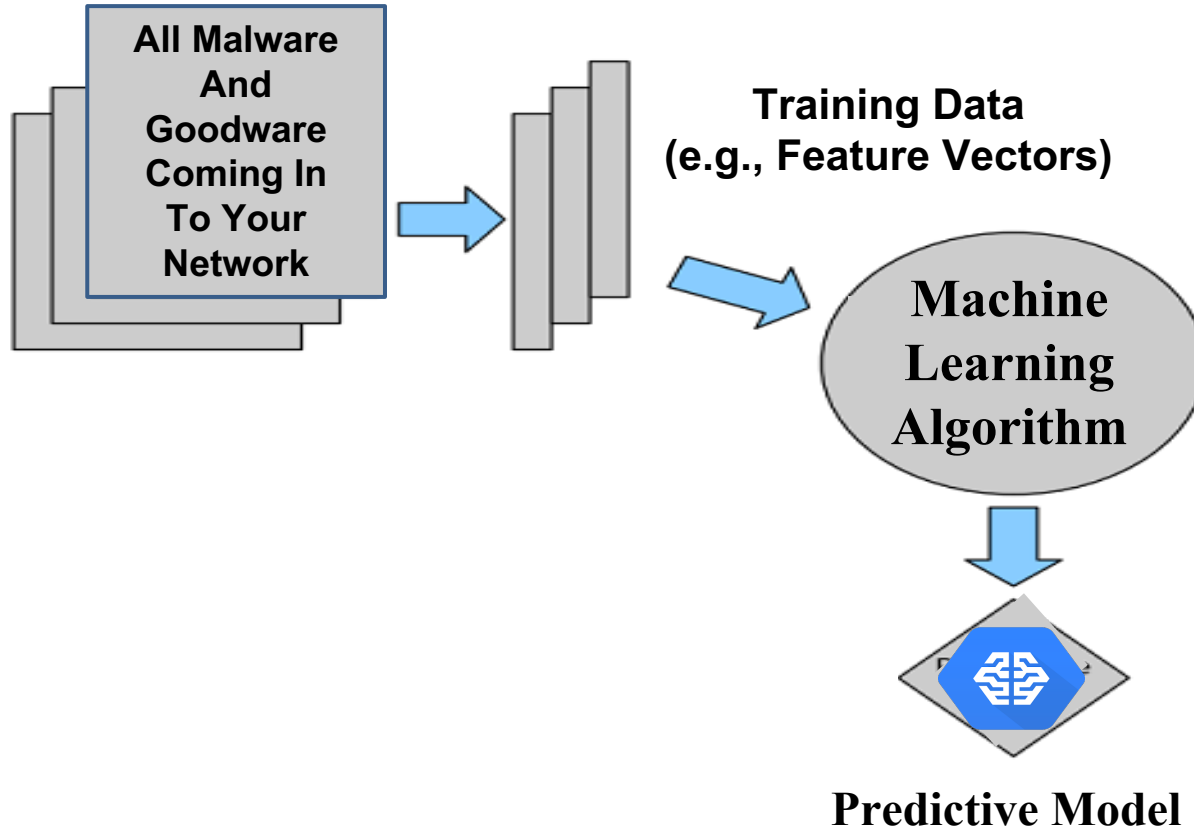
https://en.wikipedia.org/wiki/Machine_learning



**Learn a good
separation between
Malware and Goodware**



How Does Machine Learning Work?

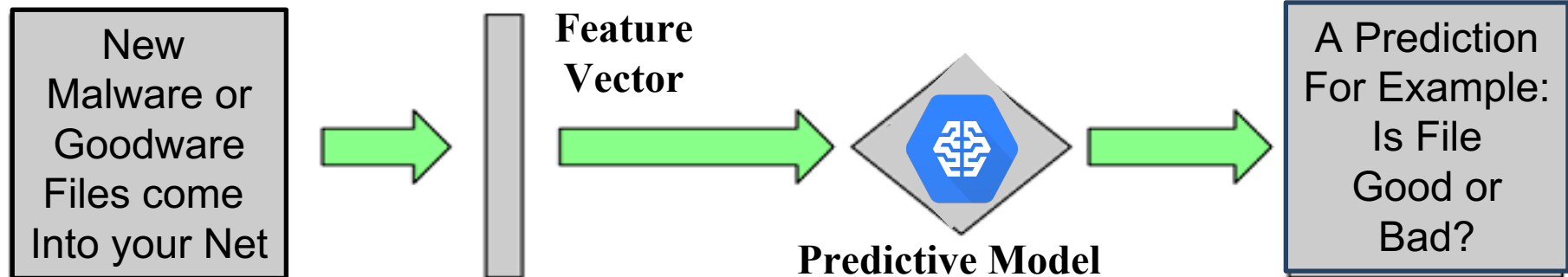


**Step 1:
Train
Predictive
Model**

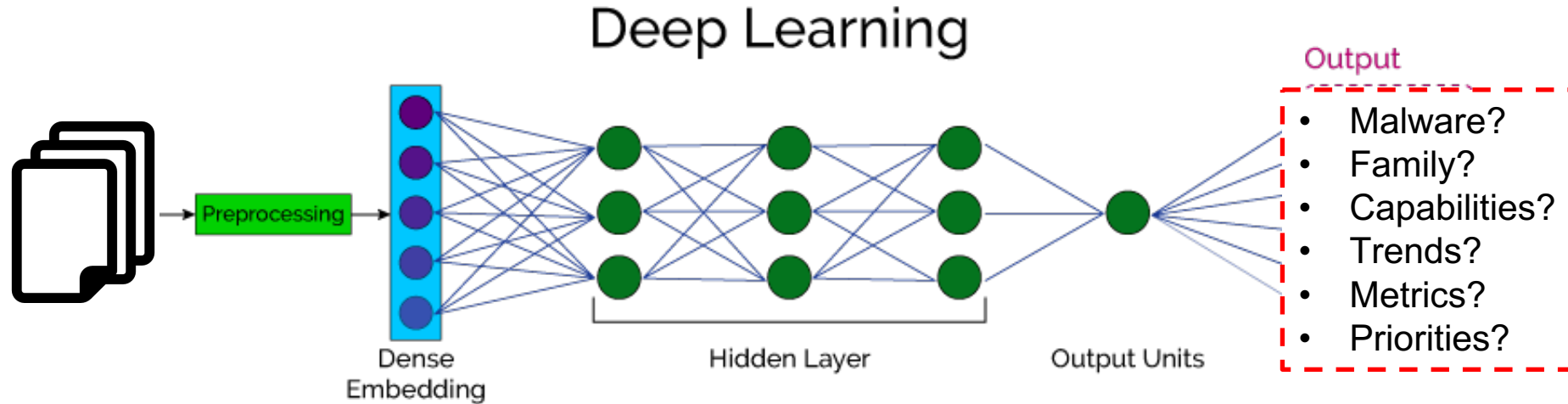


How Does Machine Learning Work?

Step 2: Deploy Predictive Model



Deep Learning is a subset of Machine Learning

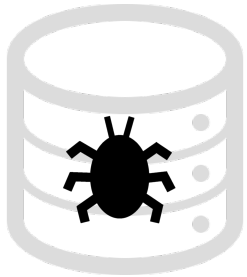


Deep Learning is a neural network with many layers of nonlinear processing units. Each successive layer uses the output from the previous layer as input.

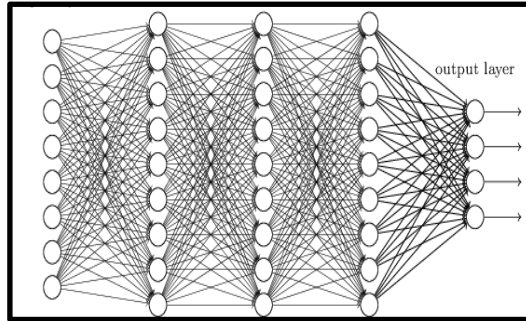


The Solution: Deep Learning + Security

Why Now? Because these are readily available!



Repository of
Billions of
Malware
Big Data

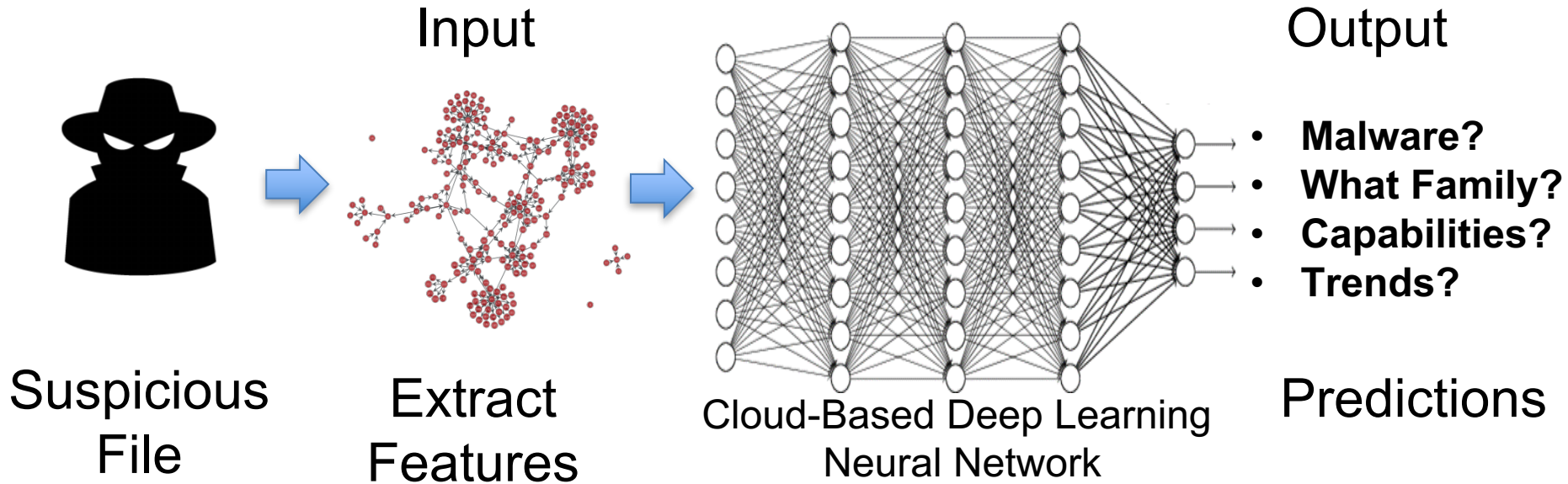


Deep Learning
Open Source
Frameworks



High-
Performance
Cloud Computing
Inexpensive

How? Deep Learning in the Cloud



Deep Learning models can predict malware with very high accuracy and at network speed



What is Malware Characterization?

```
[0x0804d638] 3x 185 /132da> pd $r @ main+2488 # 0x0804d638
; JMP XREF from 0x0804d638 (unk)
0x0804d638 8b5acc2ffff mov eax, dword [ebp - 0x3d54]
0x0804d639 89442408 mov dword [esp + 8], eax
0x0804d63a 8b45dc mov eax, dword [ebp - 0x24]
0x0804d63b 89442404 mov dword [esp + 4], eax
0x0804d63c 8d85bef3ffff lea eax, [ebp - 0xc42]
0x0804d63d 890424 mov dword [esp], eax
0x0804d63e e87afffff call sym.writefile ;[1] isyn.writefile()
0x0804d63f 85c0 test eax, eax
0x0804d640 741e je 0x0804d6fc ;[2]
0x0804d641 8d85bef3ffff lea eax, [ebp - 0xc42]
0x0804d642 890424 mov dword [esp], eax
0x0804d643 e819baffffff call sym.randads ;[3] isyn.randads()
0x0804d644 8d85bef3ffff lea eax, [ebp - 0xc42]
0x0804d645 890424 mov dword [esp], eax
0x0804d646 e87cb8ffff call sym.linuxExec
0x0804d647 e980000000 jmp 0x0804d6fc
; JMP XREF from 0x0804d638 (unk)
0x0804d648 8b5acc2ffff mov eax, dword [ebp - 0x3d54]
0x0804d649 89442408 mov dword [esp + 8], eax
0x0804d64a 8b45dc mov eax, dword [ebp - 0x24]
0x0804d64b 89442404 mov dword [esp + 4], eax
0x0804d64c 8d85bef3ffff lea eax, [ebp - 0x1042]
0x0804d64d 890424 mov dword [esp], eax
0x0804d64e e87afffff call sym.writefile ;[1] isyn.writefile()
0x0804d64f 85c0 test eax, eax
0x0804d650 741e je 0x0804d6bd ;[6]
0x0804d651 8d85bef3ffff lea eax, [ebp - 0x1042]
0x0804d652 890424 mov dword [esp], eax
0x0804d653 e819baffffff call sym.randads ;[3] isyn.randads()
0x0804d654 8d85bef3ffff lea eax, [ebp - 0x1042]
0x0804d655 890424 mov dword [esp], eax
0x0804d656 e87cb8ffff call sym.linuxExec
0x0804d657 eb3f jmp 0x0804d6fc
; JMP XREF from 0x0804d638 (unk)
0x0804d658 8b5acc2ffff mov eax, dword [ebp - 0x3d54]
0x0804d659 89442408 mov dword [esp + 8], eax
0x0804d65a 8b45dc mov eax, dword [ebp - 0x24]
0x0804d65b 89442404 mov dword [esp + 4], eax
0x0804d65c 8d85bef3ffff lea eax, [ebp - 0x1042]
0x0804d65d 890424 mov dword [esp], eax
0x0804d65e e87afffff call sym.writefile ;[1] isyn.writefile()
0x0804d65f 85c0 test eax, eax
0x0804d660 741e je 0x0804d6fc
0x0804d661 8d85bef3ffff lea eax, [ebp - 0x1442]
0x0804d662 890424 mov dword [esp], eax
0x0804d663 e819baffffff call sym.randads ;[3] isyn.randads()
0x0804d664 8d85bef3ffff lea eax, [ebp - 0x1442]
0x0804d665 890424 mov dword [esp], eax
0x0804d666 e811b7ffff call sym.linuxExec
; XREFS: JMP 0x0804d638 JMP 0x0804d638 JMP 0x0804d638
; XREFS: JMP 0x0804d677 JMP 0x0804d6bb JMP 0x0804d6de
0x0804d667 8b45d0 mov eax, dword [ebp - 0x30]
0x0804d668 89442404 mov dword [esp + 4], eax
0x0804d669 c70424020000 mov dword [esp], 2 ; checks if it runs..
0x0804d66a e860cffff call sym.HidePidPort ;[1] isyn.HidePidPort()
0x0804d66b e8dc9a0100 call sym.getpid ;[2] isyn.getpid() ; sym.__getpid
0x0804d66c 89442404 mov dword [esp + 4], eax
0x0804d66d c70424020000 mov dword [esp], 2
0x0804d66e e84bcffff call sym.HidePidPort ;[1] isyn.HidePidPort()
0x0804d66f 8d85c8f7ffff lea eax, [ebp - 0x838]
0x0804d670 890424 mov dword [esp], eax
0x0804d671 e8bee50000 call sym.remove ;[1] isyn.remove()
0x0804d672 c78544c2ffff mov dword [ebp - 0x3dbc], 0
0x0804d673 e97e090000 jmp 0x0804d638 ; cleaning up, retn & goto next process
```

- Example of Malware Characteristics:
 - Is it file encrypted?
 - Does it send spam?
 - Does it read / steal private information?
 - Will it encrypt data?
 - Does it make a network connection?
 - What does the reverse-engineered code look like?



Malware Characterization

Bytes

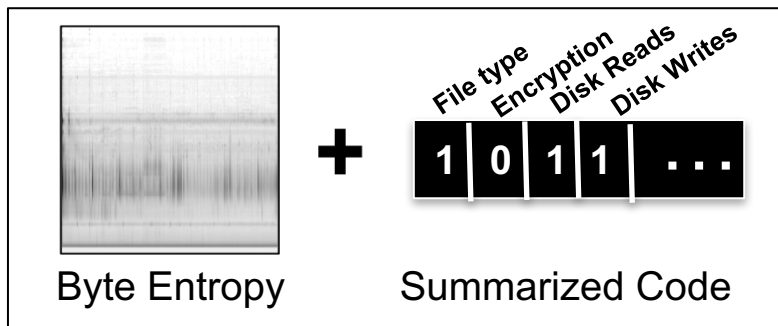
- Shannon Entropy
- Bytes N-grams
- Strings

Code

- Instruction N-grams
- Statistics
 - Function, Blocks
 - Calls, Branches

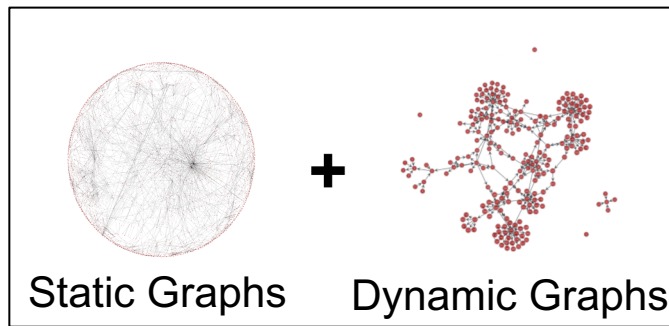
Malware Characterization

State-of-the-Art



- Feature Vector Characterization
 - Byte Entropy Histograms
 - Summarized Code Structure

Graph-Based



- Graph-Based Characterization
 - Static Code Graphs
 - Dynamic Behavioral Graphs



Malware Classification using Machine Learning





Malware Stream

Data from Reversing Labs

- Billions of malware
- Curated streams

Financial stream

- 1.65 millions
- 40+ families

Selected 11 families with > 1000 malware

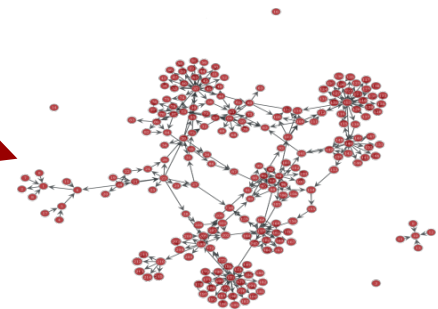
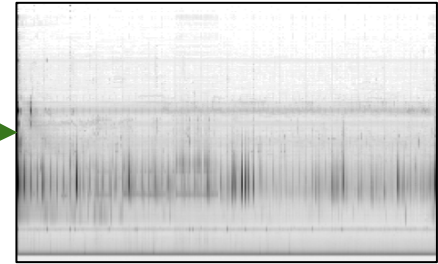


Malware Features for Deep Learning

State-of-the-Art

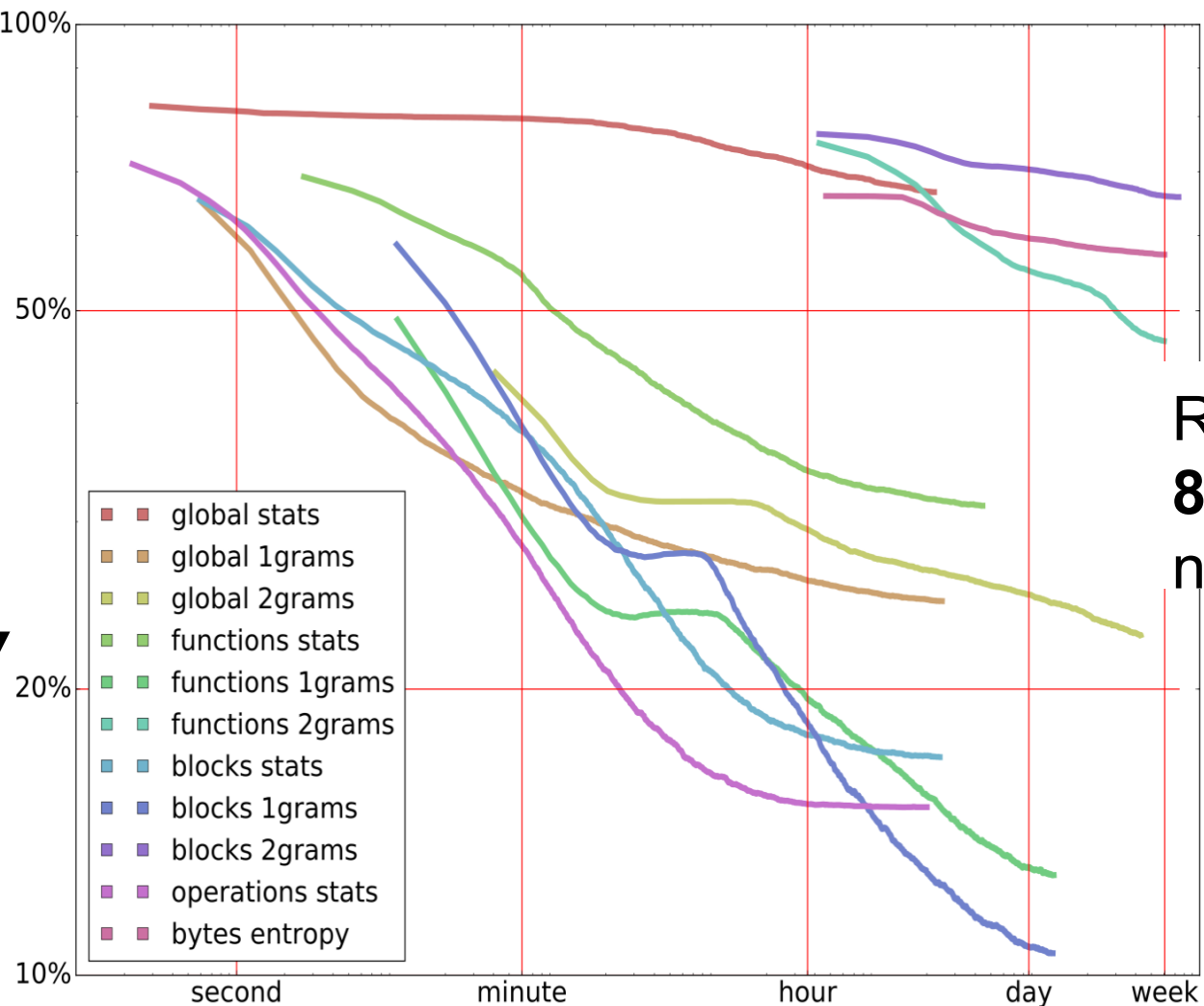
Graph-Based

Characterization		Format	Size
bytes-entropy histogram		matrix	256 x 256
Global	statistics	vector	43
	1-grams		53
	2-grams		2809
Function	statistics	matrix	20 x 23
	1-grams		20 x 53
	2-grams		20 x 2809
Block	statistics	matrix	20 x 10
	1-grams		20 x 53
	2-grams		20 x 2809
Operations	statistic	matrix	20 x 2





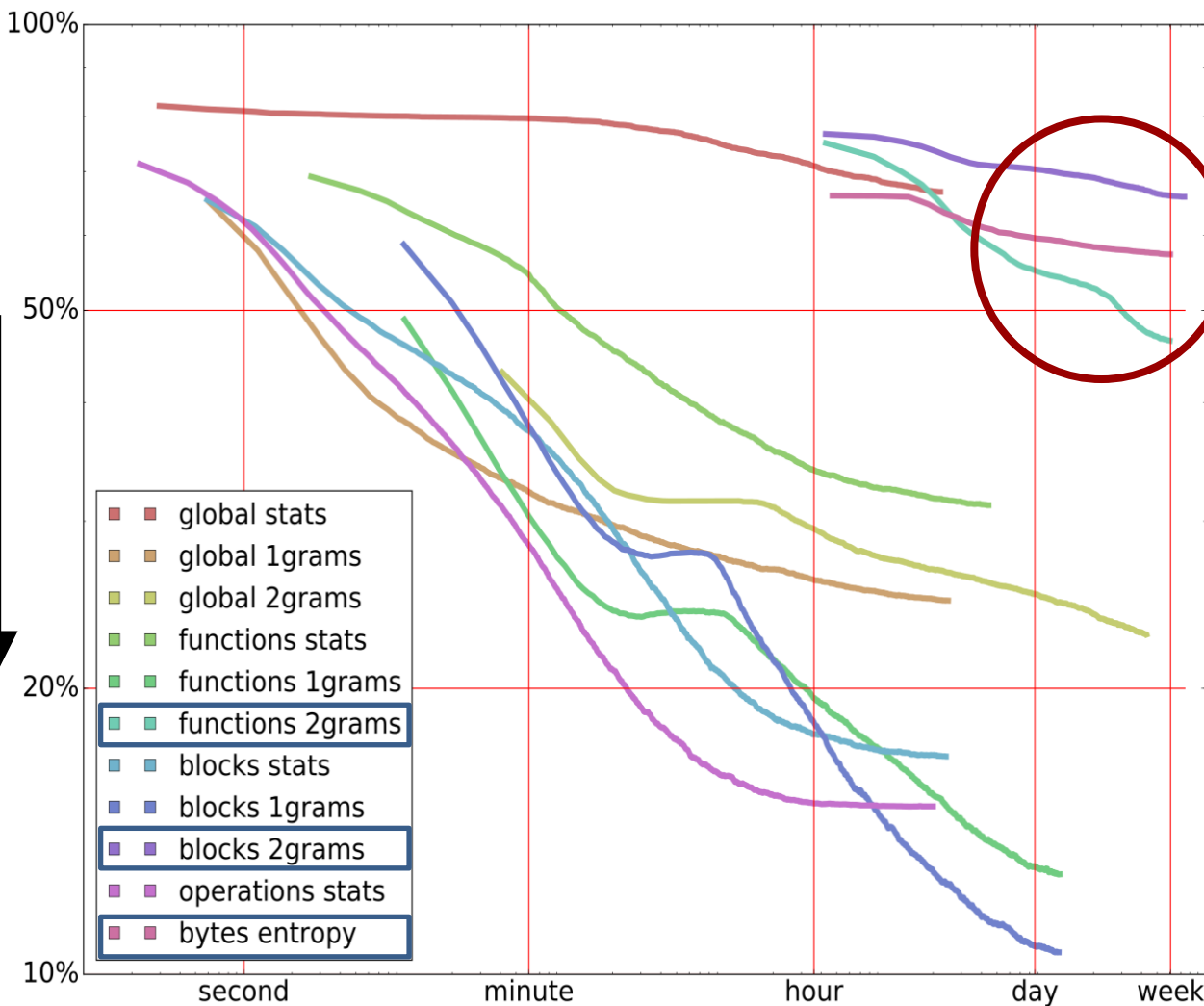
Lower is Better



Results running on
8000 compute
nodes for **2 weeks**



Lower is Better

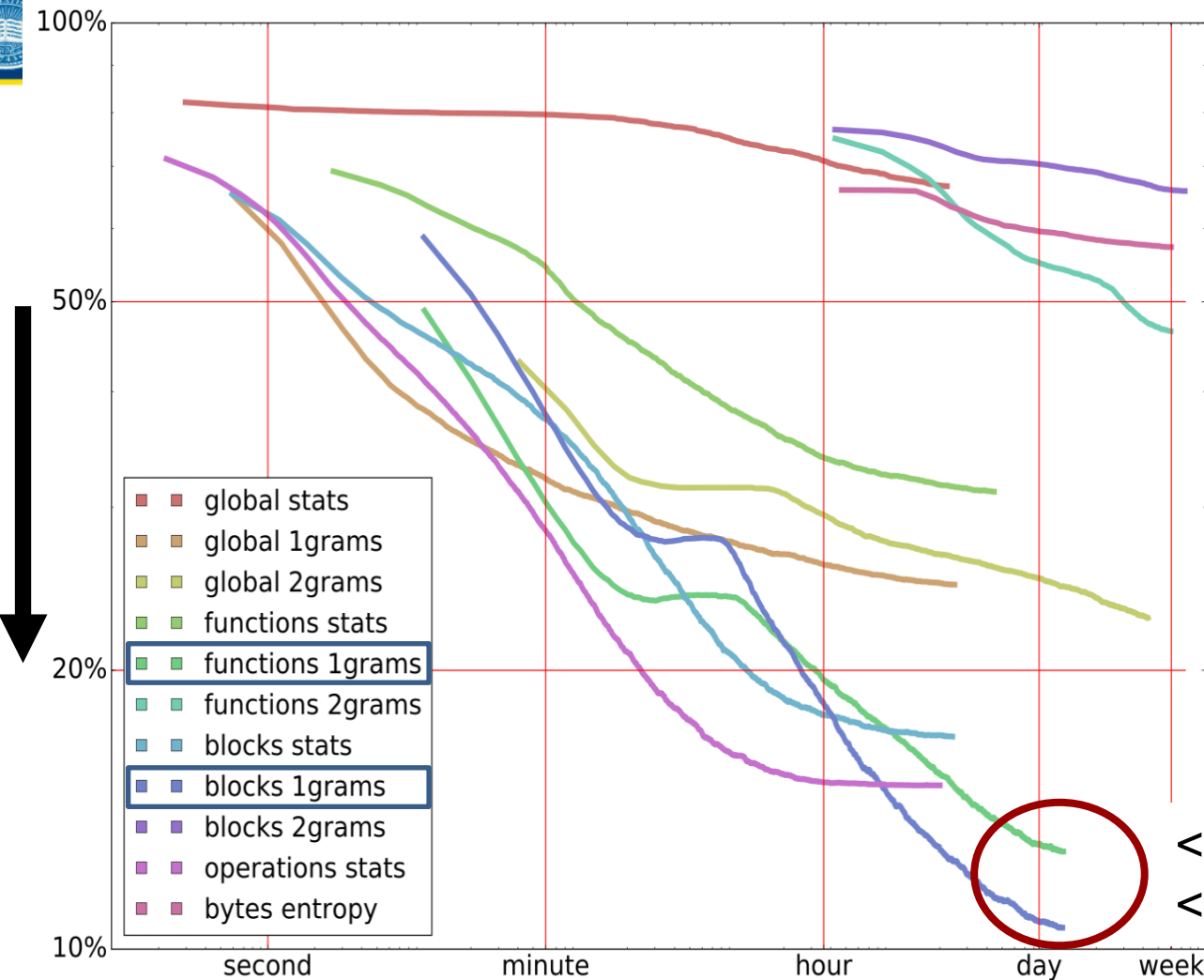


Characterization too large

- Blocks 2grams
Size: 20 x 2809
- Byte entropy
Size: 256 x 256
- Functions 2grams
Size: 20 x 2809



Lower is Better



Graph-Based

- Functions 1grams
Size: 20 x 53
- Blocks 1grams
Size: 20 x 53

< 13%

< 11%



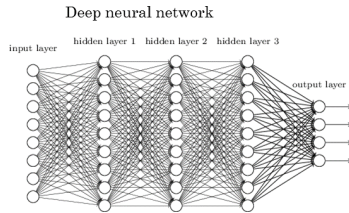
Summary of Best Results: Ensemble of DNNs

Models	1 best	2 best	5 best
BEH & Global level	16.0%	15.3%	13.8%
Graph-Based	8.4%	8.0%	8.0%
All Features	6.9%	6.5%	6.3%

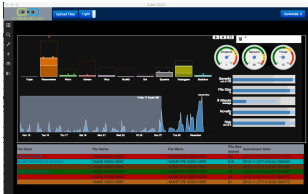
50% improvement in accuracy using Graphs!



An all-software product consisting of:



- 1) a deep learning neural network in the cloud
- 2) a network tap (“threat analytics platform”) used to extract and examine all files from traffic entering a critical infrastructure network;



- 3) An analytics user interface to view results and provide visibility;



- 4) Optional: Threat Intelligence feed into deep learning “engine.”



Questions ?



CYBER 20/20
Deep Machine Learning
Meets Cybersecurity

John Cavazos
CEO and Founder
cavazos@cyber2020.com
302-690-6041

WWW.CYBER2020.COM